CLAIMS

We claim:

1. A method, comprising:
partitioning a cache array dynamically based upon requests for memory from an integrated device having a plurality of processors.

2. The method as claimed in claim 1, further comprising subdividing one or more ways within the cache array.

3. The method as claimed in claim 1, further comprising subdividing one or more sets within the cache array.

4. The method as claimed in claim 1, further comprising using a single least recently used array to replace ways.

5. The method as claimed in claim 1, further comprising applying a multiple pseudo least recently used update based on an entry hit.

6. The method as claimed in claim 1, further comprising partitioning dynamically the cache array into a direct-mapped cache.

7. A device comprising:

a cache memory array dynamically partitioned when multiple memory requests are received from an integrated device having a plurality of processors.

8. The device as claimed in claim 7 further comprising:
an integrated device having a plurality of processors connected to the cache memory array.

9. The device as claimed in claim 7 further comprising a main memory device connected to the cache memory array.

10. The device as claimed in claim 8 wherein the integrated device includes a graphics processor and a central processing unit.

11. A computer-readable medium having stored thereon a plurality of instructions, said plurality of instructions when executed by a computer, cause said computer to perform the method of:
partitioning a cache array dynamically based upon requests for memory from an integrated device having a plurality of processors.

12. The computer-readable medium of claim 11 having stored thereon additional instructions, said additional instructions when executed by a computer, cause said computer to further perform the method of subdividing one or more ways within the cache array.

13. The computer-readable medium of claim 11 having stored thereon additional instructions, said additional instructions when executed by a computer, cause said computer to further perform the method of subdividing one or more sets within the cache array.

14. The computer-readable medium of claim 11 having stored thereon-additional instructions, said additional instructions when executed by a computer, cause said computer to further perform the method of using a single least recently used array to replace ways.

15. The computer-readable medium of claim 11 having stored thereon-additional instructions, said additional instructions when executed by a computer, cause said computer to further perform the method of applying a multiple pseudo least recently used update based on an entry hit.

16. The computer-readable medium of claim 11 having stored thereon-additional instructions, said additional instructions when executed by a computer, cause said computer to further perform the method of partitioning dynamically the cache array into a direct-mapped cache.

17. A method, comprising:

converting an N-way set associative cache dynamically into a direct mapped cache; including

removing M least significant bits from a tag address, and

adding the M least significant bits to M most significant bits of a set address of the direct-mapped cache.

18. The method of claim 17, wherein N equals 2 to the power M.

19. A method, comprising:

converting an N-way set associative cache dynamically into a Z x N-way set associative cache; including

providing Y+1 virtual copies of a pseudo-LRU array for the N-way set associative cache, wherein the pseudo-LRU array is not replicated, and

selecting a virtual copy with Y most significant bits of a set address for the N-way set associative cache.

20. The method of claim 19, wherein Z is 2 to the power Y, where Y is greater than or equal to 1.

21. The method of claim 19, wherein the Y most significant bits of the set address for the N-way set associative cache become the Y least significant bits of the tag address for the Z x N-way set associative cache.